REMARKS

This paper is submitted in reply to the Office Action dated January 11, 2007, within the three-month period for response. Reconsideration and allowance of all pending claims are respectfully requested.

In the subject Office Action, claims 1-22 were objected to by the Examiner. Furthermore, claims 21-22 were rejected under 35 U.S.C. § 101. Additionally, claims 1-5, 8-15 and 18-22 were rejected under 35 U.S.C. § 103(a) as being unpatentable over U.S. Patent No. 7,131,112 to Bartz et al. in view of U.S. Patent Application Publication No. 2003/0167446 by Thomas; claims 6-7 and 16-17 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Bartz et al. in view of Thomas and further in view of U.S. Patent No. 6,385,768 to Ziebell.

Applicants respectfully traverse the Examiner's rejections to the extent that they are maintained. Applicants have canceled claim 22 and amended claims 1-2, 11-12 and 21. Applicants respectfully submit that no new matter is being added by the above amendments, as the amendments are fully supported in the specification, drawings and claims as originally filed.

Now turning to the subject Office Action, and initially to the objection to claims 1-22, the Examiner will note that Applicants have amended claims 1-2, 11-12 and 21 to insert the word "representation" after "canonically-parsed" as suggested by the Examiner. Reconsideration and withdrawal of the objections to claims 1-21 are therefore respectfully requested.

Next with regard to the §101 rejection of claims 21 and 22, the Examiner will note that Applicants have amended claim 21 to recite a "recordable" medium, and have canceled claim 22 without prejudice. As discussed at page 16, lines 1-3, a recordable medium is a signal bearing medium of a type exemplified by "volatile and non-volatile memory devices, floppy and other removable disks, hard disk drives, magnetic tape, optical disks (e.g., CD-ROMs, DVDs, etc.), among others," all of which constitute examples of physical, tangible articles or objects. Applicants have also clarified that the program code is executable, and that the recordable medium is computer readable, as

suggested by the Examiner. While Applicants maintain that claim 21 in its original form is fully compliant with §101, Applicants respectfully submit that claim 21 as amended now additionally recites subject matter that is considered statutory according to the Office's current interpretation of its own guidelines, and withdrawal of the §101 rejection of claim 21 is therefore respectfully requested.

Now turning to the art-based rejections, and specifically to the rejection of independent claim 1, this claim generally recites a method for adapting a standard code base. The method comprises canonically parsing a modified version of a first release of a standard code base to generate a canonically-parsed representation of the modified version, generating difference data representative of changes made to the first release of the standard code base using the canonically-parsed representation of the modified version, and using the difference data in applying the changes made to the first release of the standard code base to a second release of the standard code base.

In rejecting claim 1, the Examiner relies principally on Bartz, arguing that the reference, and in particular, Figs. 2-4 and 7, col. 4, lines 29-31, col. 5, lines 10-35, col. 6, line 61 to col. 7, line 4, col. 8, line 60 to col. 9, line 13 and col. 9, lines 31-34 thereof, discloses all of the features of claim 1 with the exception of canonical parsing. However, Bartz additionally fails to disclose a number of other features of claim 1.

In particular, claim 1 recites the combination of generating difference data representative of changes made in a modified version of a first release of a standard code base, and then using the difference data to apply the changes made in the first release of the standard code base to a second release of that code base. Put another way, the difference data is not a comparison of first and second releases of a standard code base, it is instead based on a comparison of a modified version of the first release of the standard code base, which is separate from the second release of the standard code base to which the changes are applied.

An illustrative example of one possible use of the invention of claim 1 is described at page 9, lines 8-16 of the Application. Consider, for example, the standard release of the Java Development Toolkit, and in particular, the windowing package used

therein (referred to as the Java Abstract Window Toolkit(AWT)), developed by Sun Microsystems, the licensor of Java. Developers are allowed to modify this code base for their own particular needs. For example, it might be desirable to specifically adapt the AWT, which is intended for use in displaying windows on a local computer upon which a Java program is running, for use in a client/server environment where the window that may be displayed by a program running on a server is displayed on a client computer. If, for example, Sun Microsystems releases version 1.0 of the AWT, and a developer adapts this release to support a client/server environment, certain changes will have been made to that standard release to support the different functionality.

Later, Sun Microsystems updates the AWT to version 2.0, and the developer wishes to update their own customized version of the AWT to support the newer standard release, the developer will be required to make many of the same changes to the version 2.0 release as were made to the original 1.0 release. Conventionally, however, a developer would be required to manually go back through the original customized version to locate the changes made to the version 1.0 release, and then manually attempt to recreate those changes in the version 2.0 release.

Using the methodology recited in claim 1, however, a canonical parser may be used to generate difference data representative of the changes made by the developer to the version 1.0 release. This difference data may then be used to apply the same changes to the version 2.0 release. While in many cases not all changes made to the version 1.0 release will be able to be made in exactly the same manner in the version 2.0 release, the developer often has a "head start" on how to modify the version 2.0 release to enable the customized functionality that had previously been added to the version 1.0 release.

The Examiner asserts that Bartz discloses generating difference data representative of changes made to a first release of a standard code base in a modified version of the first release in Figs. 3-4 and at col. 6, line 61 to col. 7, line 4 and col. 9, lines 31-34. None of these passages, however, specifically discloses identifying changes between a particular release of a standard code base and a modified version of that release. Figs. 3 and 4 generally disclose the basic concept of comparing two documents,

but there is no disclosure in either figure of changes being made to a release of a standard code base. The passages at cols. 6-7 and 9, while arguably disclosing changes made to a particular build of a project, address how to reconcile and potentially merge changes made by different developers to the same basic file. The differences, in this case, are between different modifications to a common file.

The Examiner also asserts that Bartz discloses using the difference data in applying the changes made to the first release of the standard code base to a second release of the standard code base, citing Fig. 7 and col. 8, line 60 to col. 9, line 13. The cited passage, however, does not disclose or suggest applying changes made to one release of a code base to another release of that code base. The cited passages merely discuss determining changes that have been made to a specification, and then deciding whether to apply those changes to the same specification. Again, this relates back to the situation where multiple developers may make incompatible changes to a common file, with the routine of Fig. 7 enabling selected changes by different developers to be accepted and merged into the common file.

There is, however, no disclosure in these passages that difference data representative of the changes made to one release of a standard code base can be used to apply those changes to another release of that standard code base.

Bartz additionally discloses in Figs. 8-9, and in cols. 9-10, a concurrent building and development process where reference builds of a project are created on a periodic basis. Individual developers are able to retrieve copies of files into local enlistment areas and make changes to those files. When the files are later retrieved from a common area, the builder's local changes are applied to those files for use by the developer. There is no disclosure, however, of applying changes made in one release of a standard code base to another release of that standard code base, as recited in claim 1. The changes, when applied to a particular file, are applied to the same version to which the changes were originally applied by the developer. Accordingly, this additional embodiment of Bartz likewise fall short of disclosing this feature of claim 1.

Bartz also does not disclose a canonical parser, as has been admitted by the Examiner. In addition, in contrast to the Examiner's assertion, Bartz also does not disclose "parsing a modified version of a first release of a standard code base to generate a canonically-parsed representation of the modified version." The passages cited by the Examiner, in Fig. 3 and at col. 5, lines 10-35, discuss only searching through two documents by section, line, and character, there is no disclosure of generating a "canonically-parsed representation." In fact, the cited passage at col. 5, lines 10-35 recognizes that lines of text often correspond to semantic elements in program code, but it is clear from the passage that these semantic elements are not specifically parsed out. Bartz in effect admits that the system itself cannot parse out specific semantic elements.

As disclosed at page 10 of the Application, a canonically-parsed representation is a defined structure having a defined semantic ordering. Fig. 6 of the instant Application, and the accompanying disclosure at page 20, also illustrates one exemplary routine for generating a canonically-parsed representation, where a canonical data structure is created to represent the semantic components in the modified version of the code base. There is simply no disclosure of any similar parsing operation disclosed in Bartz. Bartz therefore also falls short of disclosing this additional feature of claim 1.

Thomas adds little to the Examiner's rejection. The Examiner cites Thomas for disclosing a canonical parsing, and irrespective of whether the reference does disclose canonical parsing, the reference does not remedy the other shortcomings of Bartz. First, Thomas, like Bartz does not disclose the use of difference data representative of changes in a modified version of a first release of a standard code base to apply those changes to a second release of that same standard code base. Thomas, in particular, is directed to detecting and tracking changes in markup language files. There is no disclosure or suggestion in the reference of software releases, or of tracking changes in modified versions of software releases, or of applying changes made to one software release to another software release.

Second, the cited passage in Fig. 2 and at paragraph [0039] discloses determining whether changes are semantic or not; however, this disclosure falls short of disclosing the

generation of a canonically-parsed representation, as required by claim 1. As noted above, the generation a canonically-parsed representation, as recited in claim 1, is more than merely detecting semantic changes in a document. A canonically-parsed representation of a code base, as recited in claim 1, is a defined structure having a defined semantic ordering of semantic elements in a code base. Thomas has no such analogous functionality, and as such, does not remedy the shortcomings of Bartz with respect to this claimed feature.

Accordingly, the combination of Bartz and Thomas does not disclose or suggest either the use of difference data associated with changes made to a first release of a standard code base to apply changes to a second release of that standard code base, or the generation of a canonically-parsed representation of a modified version of a release of a standard code base.

Applicants also submit that one of ordinary skill in the art would not be motivated to modify Bartz to incorporate either the application of changes from one release of a standard code base to another release of the code base, or the generation of a canonically-parsed representation of a modified versions of a standard code base release, as required by claim 1. Neither Bartz nor Thomas suggests either such modification, and the Examiner has presented no objective evidence of a motivation elsewhere in the prior art of record. Short of any objective evidence of a motivation in the art to make such modifications, the rejection is necessarily reliant on hindsight reconstruction, and a *prima facie* case of obviousness cannot be found.

Applicants therefore respectfully submit that since the cited references fail to disclose or suggest each and every feature of claim 1, whether considered alone or in combination, claim 1 is non-obvious over Bartz, Thomas, and the other prior art of record. Reconsideration and allowance of claim 1, and of claims 2-10 which depend therefrom, are therefore respectfully requested.

Next with regard to independent claims 11 and 21, each of these claims recites in part program code configured to canonically parse a modified version of a first release of a standard code base to generate a canonically-parsed representation of the modified

version, generate difference data representative of changes made to the first release of the standard code base using the canonically-parsed of the modified version, and use the difference data in applying the changes made to the first release of the standard code base to a second release of the standard code base. As discussed above in connection with claim 1, this combination of features is not disclosed or suggested by the prior art of record. Claims 11 and 21 are therefore non-obvious over the prior art of record for the same reasons as presented above for claim 1. Reconsideration and allowance of claims 11 and 21, and of claims 12-20 which depend therefrom, are therefore respectfully requested.

As a final matter, Applicants traverse the Examiner's rejections of the dependent claims based upon their dependency on the aforementioned independent claims. Nonetheless, Applicants do note that a number of these claims recite additional features that further distinguish these claims from the references cited by the Examiner. For example, claims 2 and 12 recite canonically parsing an unmodified version of a release of a standard code base, while claims 3 and 13 recite canonically parsing an intermediate version of a release of a standard code base. Claims 4 and 14 characterize the intermediate version as being generated by automated source transformation, with additional manual changes being made to the intermediate version to arrive at the modified version. Neither of Bartz and Thomas relates to releases of standard code bases, and as such, the references fall short of disclosing these features of claims 2-4 and 12-14.

In summary, Applicants respectfully submit that all pending claims are novel and non-obvious over the prior art of record. Reconsideration and allowance of all pending claims are therefore respectfully requested. If the Examiner has any questions regarding the foregoing, or which might otherwise further this case onto allowance, the Examiner may contact the undersigned at (513) 241-2324. Moreover, if any other charges or credits

are necessary to complete this communication, please apply them to Deposit Account 23-3000.

Respectfully submitted,

April 11, 2007
Date

/Scott A. Stinebruner/
Scott A. Stinebruner
Reg. No. 38,323
WOOD, HERRON & EVANS, L.L.P.
2700 Carew Tower
441 Vine Street
Cincinnati, Ohio 45202
Telephone: (513) 241-2324
Facsimile: (513) 241-6234